

⑨ 日本国特許庁 (JP)

⑩ 特許出願公報

⑪ 公開特許公報 (A) 昭61-133439

⑫ Int. Cl.
G 06 F 9/38

識別記号
B-7361-5B

⑬ 公開 昭和61年(1986)6月20日

審査請求 未請求 発明の数 1 (全7頁)

⑭ 発明の名称 命令先取り制御方式

⑮ 特願 昭59-256086

⑯ 出願 昭59(1984)12月4日

⑰ 発明者 桑田 明 東京都港区芝5丁目33番1号 日本電気株式会社内
⑱ 出願人 日本電気株式会社 東京都港区芝5丁目33番1号
⑲ 代理人 弁理士 内原 晋

明細書

(産業上の利用分野)

本発明は電子計算機に關し、特に命令の先取り制御方式に関する。

(従来の技術)

コンピュータの高速化技術の一つとして命令の先取り制御方式がある。ノイマン型コンピュータの特徴の一つに、次に実行される命令は、現在実行中の命令の次の命令である確率が非常に高いことが挙げられる。命令の先取り制御方式とは、この特徴を利用して外部バスの空き時間に、シーケンシャルに命令フェッテを行い、予めコンピュータ内部の命令バッファに蓄えておくことで、命令フェッテに要する時間を減らす目的としている。

第4図は従来の命令先取り機構を有するCPUのブロック図である。

CPU2は外部バス1との間でデータの授受を行う。フェッテ要求信号線119がハイレベルであればフェッテ要求信号線119を入力とするバス制御部113は外部メモリに対し命令フェッテ

1. 発明の名称
命令先取り制御方式

2. 特許請求の範囲

情報処理装置の命令先取り方式において、先取りした命令を記憶しておく記憶手段と、先取りした命令が前記記憶手段に記憶される前に先取りした命令の命令語長を検出する検出手段と、同じく先取りした命令が前記記憶手段に記憶される前に先取りした命令が少なくとも分歧命令であることを検出するデコード手段と、前記検出手段と前記デコード手段からの制御情報により、前記分歧命令を構成するバイトがすべて前記記憶手段に記憶されるとすぐに命令先取りの中止を要求する制御手段とを有することを特徴とする命令先取り制御方式。

3. 発明の詳細な説明

の要求を出す。フェッテ要求が出されたとき、バス制御部113はアドレス・データバス115上にフェッテすべき命令のアドレスを出力し、その結果外部メモリ内の前記アドレスに保持されている命令をアドレス・データバス115を通してバス制御部113に入力する。さらに内部データバス101を通して内部命令バッファ102に書きられる内部命令バッファ102はFIFO(First In First Out)構造になっており、フェッテされた命令の順に命令バス105を通してデコード106に入る。キュー・ライト・カウンタ103は、外部メモリからフェッテした命令を内部命令バッファ102に書き込む位置を示しており、キュー・リード・カウンタ103は内部命令バッファ102に書きられた命令をデコード106へ読み出す位置を示している。すなわち、キュー・ライト・カウンタ104とキュー・リード・カウンタ103の差が現在の内部命令バッファ102に書きされている命令の総バイト数となる。デコード106の出力はマイクロ

命令アドレスバス107を通してマイクロプログラム108のスタート書込地を知らせる。そのマイクロプログラムの実行によって実行ユニット110へ制御信号群109を出力し、実行ユニット110では、制御信号群109に従いバス制御部113を経由して外部メモリとの間でデータのリード/ライト動作等を行う。命令のフェッテはデータ・リード/ライトのない外部バス1の空き時間に内部命令バッファ102からのフェッテ要求に基いて行なわれるが、もし外部バス1が命令フェッテを行っている最中にデータ・リード/ライト要求が実行ユニット110から発生した場合は、そのリード/ライト要求は、現在実行中の命令フェッテが終了するまで待たされることになる。

フェッテ要求はキュー・リード・カウンタ103とキュー・ライト・カウンタ104を入力とするキュー制御部116によって、内部命令バッファ102に空きバッファがあると判断されたときに出され、フェッテ要求信号線117がハイレベルとなる。ただし、フェッテ要求がないときは、フ

ェッテ要求信号線117はローレベルであるものとする。ここで、命令のフェッテは、大部分がシーケンシャルに実行されるものという前提のもとでは有効であるが、ジャンプ、コール、リターン命令等のプログラムのフローを乱すような分岐命令がある場合、内部命令バッファ102に書きされていた分岐命令以降の命令をすべて消去し、また新たに分岐先の命令を再フェッテしなければならない。その結果、消去された分の命令フェッテが無効なものとなってしまう。ここで、その無効な命令フェッテがデータ・リード/ライト等のメモリアクセスの無い外部バス1の完全な空き時間に行なわれる時は問題ないが、もし無効な命令フェッテを行っている途中にメモリアクセス要求が生じた場合には、無効な命令フェッテサイクルが完了するまでメモリアクセス要求が待たされることとなり、その間CPU2は無効なフェッテのために無駄な時間を費すことになる。その結果、無駄な時間分全体としてのコンピュータの実行時間を遅くする原因となる。そこで、今までの命令先

取り制御の一方式として、分岐命令によって無駄になってしまい命令フェッテを減らすため、分岐命令を構成する命令バイトをすべてデコード106でデコードした後に以降の命令フェッテを中断する方法をとっていた。

以上説明した命令先取り制御方式は、AI・I-1・I-1・スペクトラム(IEEE Spectrum), 1979, 3月号, 28~34頁に記載されている方式である。

上記方式では、分岐命令を構成する全バイトを完全にデコードし命令フェッテの中断要求が出されるまでに無効な命令フェッテを行い、コンピュータの実行時間を遅くするという欠点を有していた。

本発明の目的は、分岐命令をより早く検出することでフェッテ動作をより早い段階で中断し無効なフェッテを最小限に抑えることによってコンピュータの高速化を実現するための命令先取り制御方式を提供することにある。

(問題点を解決するための手段)

本発明の命令先取り制御方式は、情報処理装置の命令先取り方式において、先取りした命令を記憶しておく記憶手段と、先取りした命令が前記記憶手段に記憶される前に先取りした命令の命令語長を検出する検出手段と、同じく先取りした命令が前記記憶手段に記憶される前に先取りした命令の命令語長を検出する検出手段と、同じく先取りした命令が少なくとも分岐命令であることを検出するデコード手段と、前記検出手段と前記デコード手段からの制御情報により、前記分岐命令を構成するバイトがすべて前記記憶手段に記憶されるとすぐに命令先取りの中止を要求する制御手段とを含んで構成される。

(実施例)

次に、本発明の実施例について図面を用いて説明する。

第1図は本発明の一実施例のブロック図である。この実施例は、先取りした命令を記憶しておく記憶手段としての内部命令バッファ202と、先取りした命令が内部命令バッファ202に記憶される前に先取りした命令の命令語長を検出する検

出手段としてのブリ・デコード201と、同じく先取りした命令が内部命令バッファ202に記憶される前に先取りした命令が少なくとも分岐命令であることを検出するデコード手段としてブリ・デコード201に内蔵されるデコードと、ブリ・デコード201からの制御情報により、分岐命令を構成するバイトがすべて内部命令バッファ202に記憶されるとすぐに命令先取りの中止を要求する制御手段としての2入力アンド回路218とを含んで構成される。

次に、この実施例の動作について説明する。

CPU4は外部データバス3との間でデータの授受を行い、フェッタ要求信号219がハイレベルであればバス制御部213は外部メモリに対し命令フェッタ要求を出す。フェッタ要求が出されたとき、バス制御部213はアドレス・データ・バス215を通してフェッタすべき命令のアドレスを外部バス3に出力し、その結果再びアドレス・データ・バス215を通して外部メモリ内のフェッタすべき命令を外部バス3からアドレス・

データ・バス215を通してバス制御部213に入力する。さらに、バス制御部213に入力された命令は内部データバス201を通して内部命令バッファ202に蓄えられる。内部命令バッファ202はFIFO構造になっており、フェッタされた命令の順に命令バス205を通してデコード206に入る。キュー・ライト・カウンタ204は外部メモリからフェッタした命令を内部命令バッファ202に書込むべき位置を指しており、キュー・リード・カウンタ203は内部命令バッファ202に蓄えられた命令をデコード206へ脱出位置を示している。すなわち、キュー・ライト・カウンタ204とキュー・リード・カウンタ203との差が内部命令バッファ202に蓄えられている命令の総バイト数となる。さらに、デコード206の出力は、マイクロ命令アドレスバス207を通してマイクロプログラム208のスタート番地を知らせ、そのマイクロプログラムの実行によって実行ユニット210へ制御信号群209を出力する。実行ユニット210では制御信号群

209に従いバス制御部213を通して、外部メモリとの間でデータのリード・ライト動作等を行う。命令のフェッタはデータのリード・ライト・サイクルの無い外部バス3の空き時間に、キュー制御部216からのフェッタ要求信号217に従って行なわれる。フェッタ要求はキュー・ライト・カウンタ204とキュー・リード・カウンタ203を入力とするキュー制御部216によって内部命令バッファ202に空バッファがあると判断されたときに出され、キュー制御部216からのフェッタ要求信号217がハイレベルとなり、内部命令バッファ202からのフェッタ要求を発生する。但し、フェッタ要求が無いとき、フェッタ要求信号217は、ロウレベルであるものとする。従来例でも述べたように命令フェッタは大部の命令がシーケンシャルに実行されるものという前提のもとでは有効であるが、ジャンプ・コール・リターン命令等のプログラムのフローを乱すような分岐命令がある場合には内部命令バッファ202に蓄えられていた分岐命令以降の命令をすべて消去し、

また新たに分岐先の命令を再フェッチしなければならない。そこで、本実施例では内部命令バッファ202の前段にブリ・デコード220を設け、ブリ・デコード220はフェッチした命令が分岐命令であり、かつその分岐命令を構成する命令バイトがすべて内部命令バッファ202に格納されたことを検知するとフェッチ中断信号線221をロウレベルにし、フェッチの中断を要求する。但し、通常フェッチ中断信号線221はハイレベルであるものとする。フェッチ中断信号線221がロウレベルであれば2入力アンド回路218の出力信号線219はロウレベルとなり、命令フェッチの中断をバス制御部213に知らせる。一方、フェッチ中断信号線221がハイレベルであり、かつ内部命令バッファ202に空バッファがあつてキューブ制御部216からのフェッチ要求信号線217がハイレベル(フェッチ要求)であれば、2入力アンド回路218の出力線219はハイレベルとなりバス制御部213にフェッチの要求をする。

行い、保持していた内容を-1すると同時にラッチャ回路310にデクリメントした後の内容を出力する。ここで、カウンタ309はその値が“0”になると制御信号線311をハイレベルにし、またラッチャ回路310はその値が“0”になると制御信号線312をハイレベルにする。さらにカウンタ309及びラッチャ回路310は初期状態(リセット時)において、その値は“0”になっている。バス制御部213からの制御信号線222は現在実行しているバス・サイクルがフェッチ・サイクルであるときにハイレベルとなる制御信号であり、制御信号線223は制御信号線222より1クロック分遅延してハイレベルとなる制御信号である。-2入力アンド回路-3-1-3は制御信号線-1-2-2-2及び制御信号線312を入力とし、制御信号線222がハイレベル、すなわち、フェッチ・サイクルであり、かつ制御信号線312がハイレベル、すなわちラッチャ回路310の値が“0”であるとき制御信号線314とハイレベルにし、ゲート301及びゲート302を共にオンにする。

本発明は、以上述べてきたよう構成により内部命令バッファ202の前段にブリ・デコード220を設けることで、分岐命令をより早く検知し、内部命令バッファ202に分岐命令を構成する命令バイトがすべて格納されていることを確認すると直ちにバス制御部213にフェッチの中断を要求し、無効なフェッチを最小限に抑える手段を提供するものである。

第2図は第1図に示すブリ・デコードの詳細回路図である。

デコード305は信号線303より入力したデータから現在フェッチしている命令語長を検出し、信号線307を通してカウンタ309へ出力する。デコード306は、信号線304より入力したデータから現在フェッチしている命令が分岐命令か否かを検出し、分岐命令である場合は制御信号線308をハイレベルにする。カウンタ309は信号線より現在フェッチしている命令の命令語長を入力し、バス制御部213からの制御信号線223がハイレベルになると、1回デクリメント動作を

ゲート301及び302はオン状態で内部データバス201上の値をそれぞれデータバス303及びデータバス304へ出力する。2入力アンド回路315は制御信号線311及び制御信号線308を入力とし、制御信号線311がハイレベル、すなわちカウンタ309の値が“0”であり、かつ制御信号線308がハイレベル、すなわち現在フェッチしている命令が分岐命令であるとき、制御信号線316をハイレベルにする。ラッチャ回路317は制御信号線316がハイレベルになると制御信号線221をロウレベルにし、制御信号線318がロウレベルになるまでロウレベルを出力する。制御信号線318は分岐先のアドレスが確定しフェッチの再開要求によってロウレベルになる。制御信号線318がロウレベルになると、ラッチャ回路317は制御信号線221をハイレベルにし、制御信号線316が再びハイレベルになるまで制御信号線221はハイレベルを出力している。

次に、第3図に示すタイミング図を用いて第2

図に示すプリ・デコード220の動作について説明する。ここで、リセット後最初にフェッテする命令が分岐命令でない3バイト命令であり、次にフェッテする命令が3バイトの命令Bであるものとする。

まず、リセット直後カウンタ309及びラッテ回路310は共に“0”となっており、ラッテ回路310が“0”であるため制御信号線312はハイレベルとなっている。

次に、最初の3バイト命令Aの第1バイト目のフェッテ・サイクルが起ると制御信号線222がハイレベルとなり、制御信号線222がハイレベルの間、2入力アンド回路313の出力信号線314がハイレベルとなる。制御信号線314がハイレベルのとき、ゲート301及びゲート302が共にオンとなり、内部データバス201上の第1バイト目がデータバス303及びデータバス304を通してデコード305及び306に入る。デコード305では、入力した第1バイト目から命令Aの命令語長(=3)をデコードし、信号線

307を通してカウンタ309へ命令語長(=3)を出力する。一方、デコード306では入力した第1バイト目から命令Aが分岐命令か否かを判断し、この場合は分岐命令ではないため制御信号線308はロウレベルを出力する。カウンタ309は信号線307を通して入力した命令語長(=3)を保持し、バス制御部213からの制御信号線223がハイレベルになると保持していた値をデクリメントすると共にそのデクリメントした値(=2)をラッテ回路310へ出力する。このとき、ラッテ回路310の値が“2”となるため、制御信号線312はロウレベルとなり、以後制御信号線222がハイレベルとなっても2入力アンド回路313の出力信号線314はロウレベルのままでゲート301及びゲート302はオフのため、内部データバス201上のデータはデータバス303及び304には入力されない。カウンタ309はデクリメントされた値(=2)を保持しているが、フェッテ・サイクル毎に制御信号線223がハイレベルになるため、その都度保持し

ている値を“-1”するとともにラッテ回路310へデクリメントした値を出力する。

第3図に示すように、命令Aの第3バイト目のフェッテ・サイクルで制御信号線223がハイレベルとなるとカウンタ309は保持していた値(=1)をデクリメントするとともにデクリメントした値(=0)をラッテ回路310へ出力する。ここで再びラッテ回路310の内容が“0”となるため、制御信号線312はハイレベルとなり、次の命令Bの第1バイト目のフェッテ・サイクルで制御信号線222がハイレベルとなると2入力アンド回路313の出力信号線314がハイレベルとなり、ゲート301及びゲート302がオンして、内部データバス201上の命令Bの第1バイト目をデコード305及び306に入力する。デコード305では、入力した第1バイト目から命令Bの命令語長(=2)をデコードし、信号線307を通してカウンタ309へ命令語長(=2)を出力する。一方、デコード306では、入力した第1バイト目から命令Bが分岐命令か否かを判断し、

この場合は分岐命令であったとすると、制御信号線308はハイレベルを出力する。カウンタ309は命令Bの命令語長(=2)を保持し、制御信号線223がハイレベルになると保持していた値をデクリメントするとともに、デクリメントされた値(=1)をラッテ回路310へ出力する。

次に、命令Bの第2バイト目のフェッテ・サイクルで制御信号線223がハイレベルになるとカウンタ309は保持していた値(=1)をデクリメントすると共にデクリメントされた値(=0)をラッテ回路310へ出力する。ここでラッテ310の値が“0”となるため、制御信号線311はハイレベルとなる。一方、デコード306の制御信号線308はハイレベルとなっているため、2入力アンド回路315の出力信号線316もハイレベルとなる。制御信号線316がハイレベルとなるとラッテ回路317は制御信号線221をロウレベルにする。制御信号線221は制御信号線318がロウレベルになるまでロウレベルを出力する。制御信号線221がロウレベルであるため、

フェッテ要求信号線 217 がハイレベルとなりフェッテ要求が発生したとしても制御信号線 221 がロウレベルであるため、2 入力アンド回路 218 の出力信号線 219 はロウレベルのままで、バスコントロール 213 に對しフェッテ要求を行なわない。フェッテ要求は分岐先のアドレスが確定し、制御信号線 318 がロウレベルとなって制御信号線 221 がハイレベルとなって初めて受けられる。

上記実施例のように、外部バスからプロセッサ内部に命令を取り込む前にブリ・デコードを設け、現在取り込んでいる命令が分岐命令であり、かつ分岐命令と構成する命令バイトがすべてプロセッサ内部に取り込まれたことを検出することでより早い段階でフェッテの中断要求を出力し、無効な命令フェッテを最小限に抑えることが可能となる。
(発明の効果)

以上説明したように、本発明は内部命令バッファに外部データバスより命令をフェッテする前段にブリ・デコードを設けフェッテする命令の第 1

バイト目をデコードし、デコードした命令が分岐命令である場合、分岐命令を構成する命令バイトがすべてフェッテされたことを検出すると、直ちにフェッテの中断要求を出して命令フェッテを中断し、分岐命令の実行によって無駄になってしまい命令フェッテを最小限に抑え、データアクセス待ち時間を減らし、コンピュータの高速化を計ることが可能となるという効果を有する。

4. 図面の簡単な説明

第 1 図は本発明の一実施例のブロック図、第 2 図は第 1 図に示すブリ・デコードの詳細回路図、第 3 図は第 2 図に示したブリ・デコードの動作時のタイミング図、第 4 図は従来の命令先取り機構を有する CPU のブロック図である。

1 ……外部データバス、2 ……CPU、3 ……外部データバス、4 ……CPU、101, 201 ……命令バス、102, 202 ……内部命令バッファ、103, 203 ……キュー・リード・カウンタ、104, 204 ……キュー・ライト・カウ

ンタ、105, 205 ……命令バス、106, 206 ……デコード、107, 207 ……マイクロ命令アドレスバス、108, 208 ……マイクロプロダラム、109, 209 ……制御信号群、110, 210 ……実行ユニット、111, 221 ……フェッテ中断信号線、113, 213 ……バス制御部、114, 115, 214, 215 ……アドレス・データ・バス、116, 216 ……キュー制御部、117, 217 ……フェッテ要求信号線、118, 218 ……2 入力アンド回路、119, 219 ……フェッテ要求信号線、220 ……ブリ・デコード、222, 223 ……フェッテサイクル信号線、301, 302 ……ゲート、303, 304 ……データバス、305, 306 ……デコード、307 ……命令長信号線、308 ……分岐命令信号線、309 ……カウンタ、310 ……ラッチ、311, 312 ……制御信号線、313 ……2 入力アンド回路、314 ……制御信号線、315 ……2 入力アンド回路、316 ……制御信号線、317 ……ラッチ回路、318 ……フェッテ再開信号線。

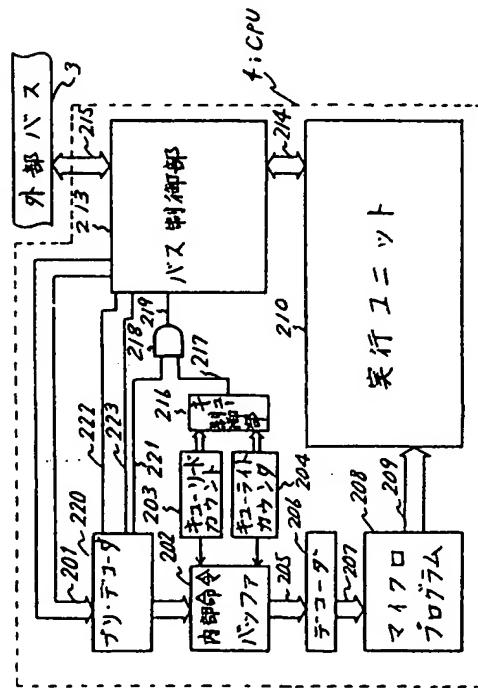
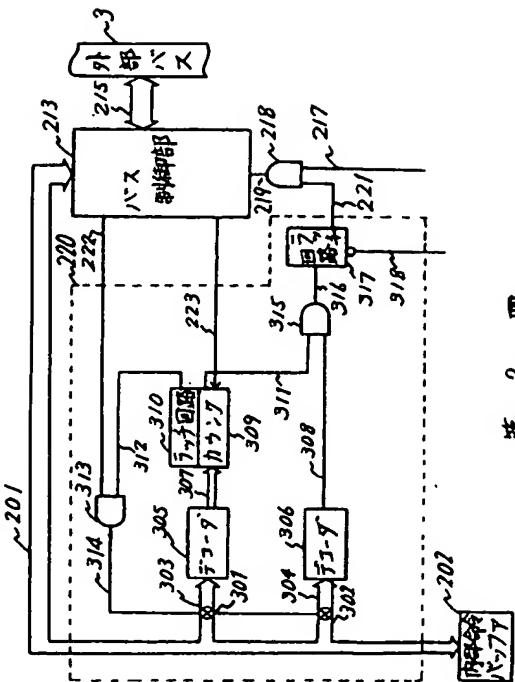
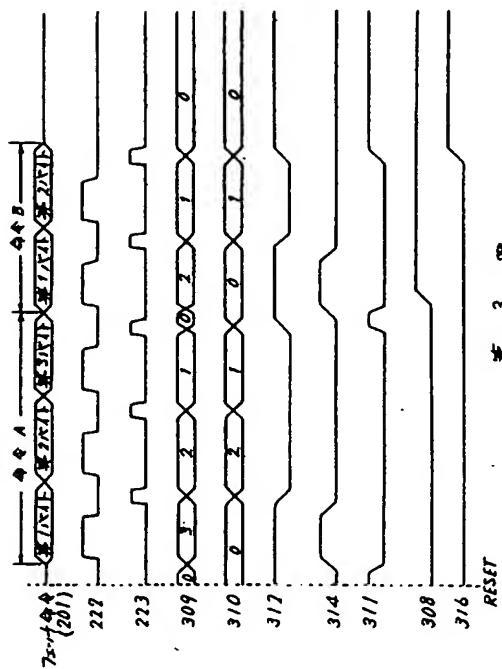


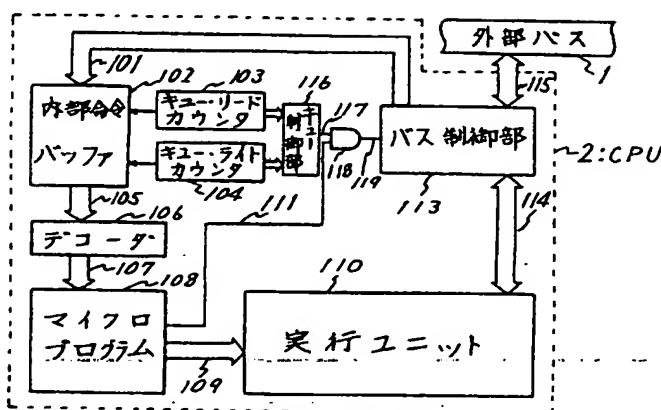
図 1



四二



四
三



第四圖

Kokai No.: S61-133439

Publication date: June 20, 1986

Application No.: S59-256086

Application date: December 4, 1984

Inventor: Kuwata Akira

Applicant: NEC Corporation

Specification

[1. Title of the Invention]

Instruction Pre-fetch Control System

[2. Claims]

[Claim 1] An instruction pre-fetch system comprising;
a storage means to store a pre-fetched instruction,
a detection means to detect instruction word length of the pre-fetched instruction before the pre-fetched instruction is stored in said storage means,
a decode means to detect the pre-fetched instruction is at least a branch instruction before the pre-fetched instruction is stored in said storage means,
a control means to request abortion of instruction pre-fetching as soon as all bytes composing said branch instruction are stored in said storage means,
according to control information from said detection means and said decode means.

[3. Detailed description of the Invention]

The present invention is related to a computer, especially related to a pre-fetch control system of an instruction.

[Prior Arts]

One of methods for increasing speed of a computer is an instruction pre-fetch control system. One of characteristics of a von Neumann computer is a high probability that an instruction to be executed in the next is an instruction follows an instruction currently being executed. The instruction pre-fetch system utilizes the characteristic to decrease a time requires for instruction fetching with performing an instruction fetching to a sequential file during a

spare time of an external bus, and storing in an instruction buffer placed in the computer beforehand.

Fig.4 is a block diagram of a CPU having a traditional pre-fetch mechanism. CPU2 exchanges data with an external bus 1. When a fetch request signal line 119 is in high-level, a bus control part 113 whose input is the fetch request signal line 119 issues an instruction fetch request to the external memory. When the fetch request is issued, the bus control part 113 outputs an address of an instruction to be fetched on an address data bus 115, as a result, an instruction stored in said address of the external memory is input to the bus control part 113 through the address data bus 115. Furthermore, an internal instruction buffer 102 stored in the internal instruction buffer 102 through an internal data bus 101 has a FIFO (First In First Out) structure and entered in a decoder 106 through an instruction bus 105 in order of fetched instructions. A queue write counter 104 indicates a position of the internal instruction buffer 102 to which an instruction fetched from the external memory is written, and a queue read counter 103 indicates a position from which an instruction stored in the internal instruction buffer 102 is read out to a decoder 106. That is, a difference between the queue write counter 104 and the queue read counter 103 is a total number of bytes currently stored in the internal instruction buffer 102. An output of the decoder 106 notifies a start address of a micro program 108 through a micro instruction address path 107. By execution of the micro program, a group of control signals 109 is output to an execution unit 110, and in the execution unit 110, read/write operation of data with the external memory through the bus control part 113 according to the group of control signals 109. Fetching of an instruction is performed based on the fetch request from the internal instruction buffer 102 in the spare time of the external bus 1 which does not have data read/write. However, if data read/write request is generated from the execution unit 110 during the external bus 1 is performing instruction fetching, the read/write request is kept waiting until the instruction fetching currently performed is completed.

The fetch request is issued when it is determined that there is a vacant buffer in the internal instruction buffer 102 by the queue control part 116 whose inputs are the queue read counter 103 and the queue write counter 104, then a fetch request signal line 117 becomes to be a high-level. However, when there is no fetch request, the fetch request signal line 117 is in

low-level. Though the instruction fetching is effective under the assumption that most of instructions are performed sequentially, when there is a branch instruction which disrupts a flow of program such as a jump, a call or a return instruction, all instructions after the branch instruction stored in the internal instruction buffer 102 have to be deleted, and instructions at branch destination have to be fetched again. As a result, deleted instruction fetch is unproductive. When the unproductive instruction fetch is performed in a perfect spare time of the external bus 1 in which there is no memory access such as data read/write, there is no problem. However, if a memory access request arises while the unproductive instruction fetch is performed, the memory access request is kept waiting until the unproductive instruction fetch cycle is completed, then the CPU 2 spent wasteful time for the unproductive fetch. As a result, the wasteful time causes delay of the execution time of the computer as a whole. Consequently, as one of traditional instruction pre-fetch controls, there is a method to abort instruction fetching after decoding all instruction bytes composing a branch instruction with the decoder 105 for reducing instruction fetch which will be unproductive after a branch instruction.

An instruction pre-fetch control system above mentioned is a system described in IEEE Spectrum, 1979, March, P28 – 34.

The method above mentioned has a defect that all bytes composing a branch instruction are perfectly decoded, unproductive instruction fetching is performed until an abort request is issued, then execution time of a computer is delayed.

The purpose of the present invention is to provide an instruction pre-fetch control system in which increasing of processing speed of a computer with minimizing unproductive fetch with aborting fetch operation in earlier stage by detecting a branch instruction earlier.

[Means to Solve the Problem]

In an instruction pre-fetch system of an information processing device, an instruction pre-fetch system according to the present invention comprising; a storage means to store a pre-fetched instruction, a detection means to detect instruction word length of the pre-fetched instruction before the pre-fetched instruction is stored in said storage means, a decode means to detect the pre-fetched instruction is at least a branch instruction before the

pre-fetched instruction is stored in said storage means, a control means to request abortion of instruction pre-fetching as soon as all bytes composing said branch instruction are stored in said storage means, according to control information from said detection means and said decode means.

[Embodiments]

Next, an embodiment of the present invention is explained with drawings.

Fig.1 is a block diagram of one embodiment of the present invention.

This embodiment comprises; an internal instruction buffer 202 as a storage means to store a pre-fetched instruction, a pre-decoder 201 as a detection means to detect instruction word length of the pre-fetched instruction before the pre-fetched instruction is stored in the internal instruction buffer 202, a decoder included in the pre-decoder 201 as a decode means to detect the pre-fetched instruction is at least a branch instruction before the pre-fetched instruction is stored in the internal instruction buffer 202, a two inputs AND circuit 218 as a means to request abortion of pre-fetching as soon as all bytes composing the branch instruction are stored in the internal instruction buffer 202 according to a control information from the pre-decoder 201.

Next, an operation of this embodiment is explained.

CPU 4 exchanges data with the external data bus 3, and when a fetch request signal 219 is in the high-level, the bus control part 213 issues an instruction fetch request to the external memory. When the fetch request is issued, the bus control part 213 outputs an address of an instruction to be fetched to the external bus 3, as a result, through the address data bus 215 again, the instruction to be fetched in the external memory from the external bus 3 to the bus control part 213 through the address data bus 215. Furthermore, the instruction input in the bus control part 213 is stored in the internal instruction buffer 202 through the internal data bus 201. The internal instruction buffer 202 has a FIFO structure, and instructions are sent to the decoder 206 through the instruction bus in order of being fetched. The queue write counter 204 points a position of the internal instruction buffer 202 to which the instruction fetched from the external memory is written, the queue read counter 203 points a position from which the instruction stored in the internal instruction buffer 202 is read out to the decoder 206. That is, the difference between the queue write counter 204 and the queue read counter 203 is a total number of bytes stored in the internal

instruction buffer 202. Moreover, an output of the decoder 205 notifies a start address of a micro program 208 through a micro instruction address bus 207, a group of control signals 209 is output to an execution unit 210 by execution of the micro program. The execution unit 210 performs read/write operation with the external memory through the bus control part 213 according to the group of control signals 209. Fetching of an instruction is performed according to a fetch request signal 217 from the queue control part 216 during a spare time of the external bus 3 where there is no read/write cycle of data. The fetch request is issued when it is determined that there is a vacant buffer in the internal instruction buffer 202 by the queue control part 316 whose inputs are the queue write counter 204 and the queue read counter 203. Then a fetch request signal 217 from the queue control part 216 becomes to be high-level, and a fetch request from the internal instruction buffer 202 is generated. Here, when there is no fetch request, the fetch request signal 217 is in low-level. As mentioned in an example of the prior arts, the instruction fetch is effective when it is assumed that most of instructions are executed sequentially. However, if there is an instruction which disrupts a flow of program such as a jump, a call or a return instruction, all instructions after the branch instruction stored in the internal instruction buffer 202 have to be deleted, and instructions at branch destination have to be fetched again. Therefore, in this embodiment, a pre-decoder 220 is placed in a step prior to the internal instruction buffer, the pre-decoder 220 requests abortion of fetching when it is detected that the fetched instruction is a branch instruction and all instruction bytes composing the branch instruction are stored in the internal instruction buffer 202, with making a fetch abortion signal line 221 in low-level. Wherein, the fetch abortion signal line 221 is usually in high-level. When the fetch abortion signal line 221 is in low-level, an output signal line 219 of a two-inputs AND circuit 218 is in low-level to notify abortion of instruction fetching to the bus control part 213. On the other hand, when the fetch abortion signal line 221 is in high-level, there is a vacant buffer in the internal instruction buffer 202, and the fetch request signal line 217 from the queue control part 216 is in high-level (fetch request), an output line 219 of the two-inputs AND circuit 215 is in high-level to request fetching to the bus control part 213.

The present invention provides a means of minimizing unproductive fetching

with detecting a branch instruction earlier, as soon as it is detected that all instruction bytes composing the branch instruction are stored in the internal instruction buffer 202, abortion of fetching is requested to the bus control part 213 by placing the pre-decoder 220 in a step prior to the internal instruction buffer 202.

Fig.2 is a detailed circuit diagram of the pre-decoder shown in Fig.1.

A decoder 305 detects instruction word length currently fetched from data input from a signal line 303, and outputs to a counter 309 through a signal line 307. A decoder 306 detects whether a currently fetched instruction is a branch instruction, when the instruction is a branch instruction, and makes the control signal line 308 in high-level. A counter 309 inputs the instruction word length of a currently-fetched instruction, when a control signal 223 from the bus control part 213 is in high-level, performs decrement operation once to decrement stored contents and outputs contents after decrement to a latch circuit 310 simultaneously. Wherein, the counter 309 makes the control signal 311 in high-level when its value is "0", and the latch circuit makes the control signal 312 in high-level when its value is "0". Furthermore, each value of the counter 309 and the latch circuit 310 is set to "0" in the initial state (at resetting). A control signal line 222 from the bus control part 213 is a control signal in high-level when a currently executed bus cycle is a fetch cycle, a control signal line 223 is a control signal which becomes in high-level 1 clock delayed compared to the control signal line 222. Inputs of the two-inputs AND circuit 313 are the control signal line 222 and the control signal line 312, when the control signal line 222 is in high-level, that is in a fetch cycle, and the control signal line 312 is in high-level, that is a value of the latch circuit 310 is "0", the control signal line is made to be in high-level to make both of the gate 301 and the gate 302 to be ON.

The gates 301 and 302 output a value on the internal data bus 201 to the data bus 303 and the data bus 304 respectively when they are ON states. Inputs of the two-inputs AND circuit 315 are the control signal line 311 and the control signal line 305, the control signal line 316 is made to be high-level when the control signal line 311 is in high-level, that is a value of the counter is "0", and the control signal line 308 is in high-level, that is the currently fetched instruction is a branch-instruction. The latch circuit 317 makes the control signal line 221 in low-level when the control signal line 316 is in high-level, and outputs low-level until the control signal line 318 becomes to

be low-level. The control signal line 318 becomes to be low-level according to a fetch re-start request when a branch destination address is determined. When the control signal line 318 becomes to be low-level, the latch circuit 317 makes the control signal line 221 in high-level, and the control signal line 221 outputs high-level until the control signal line 316 becomes in high-level again.

Next, an operation of pre-decoder 220 shown in Fig.2 is explained with a timing chart shown in Fig.3. Wherein, it is assumed that a first fetched instruction after resetting is a 3-bytes instruction A which is not a branch instruction and a second fetched instruction is a 3-bytes instruction B.

Both the counter 309 and the latch circuit 310 just after the resetting are "0", as the latch circuit 310 is "0", the control signal line 312 is in high-level.

Next, when a fetch cycle of a first byte of the first 3-bytes instruction A is started, the control signal line 222 becomes to be high-level, while the signal line 222 is in high-level, the output signal line 314 of the two-inputs AND circuit 313 is in high-level. When the control signal line 314 is in high-level, the gate 301 and the gate 302 become to be ON, a first byte on the internal data bus 201 is sent to the decoders 305 and 306 through the data bus 304. The decoder 305 decodes an instruction word length (=3) of the instruction A from the input first byte, and the instruction word length (=3) is output to the counter 309 through the signal line 307. On the other hand, the decoder 306 decides whether the instruction A is a branch instruction from the input first byte. In this case, as the instruction is not a branch instruction, the control signal line 308 outputs low-level. The counter 309 holds the instruction word length (=3) input through the signal line 307, decrements the held value and outputs the decremented value (=2) to the latch circuit 310 when the control signal line 223 from the bus control part 213 becomes to be high-level. At this time, as the value of the latch circuit 310 becomes to be "2", the control signal line 312 becomes to be low-level, after that as the output signal line 314 of the two-inputs AND circuit 313 keeps low-level and the gate 301 and the gate 302 keep OFF even if the control signal line 222 becomes to be high-level, data on the internal data bus 201 is not input to the data buses 303 and 304. Though the counter 309 keeps decremented value (=2), as the control signal line 223 becomes to be high-level at each fetch cycle, each time keeping value is decremented and the decremented value is returned to the latch circuit 310.

As shown in Fig.3, when the control signal 223 becomes to be high-level at a fetch cycle of a third byte of the instruction A, the counter 309 decrements holding value (=1) and outputs the decremented value (=0) to the latch circuit 310. As the content of the latch circuit 310 becomes "0" again, the control signal line 312 becomes to be high-level, when the control signal line 222 becomes to be high-level at a fetch cycle of the first byte of the instruction B, the output signal line 314 of the two-inputs AND circuit 313 becomes to be high-level, the gates 301 and 302 are turned to ON, a first byte of the instruction B on the internal data bus 201 is input to the decoders 305 and 306. The decoder 305 decodes the instruction word length (=2) of the instruction B from the input first byte, and outputs the instruction word length (=2) to the counter 309 through the signal line 307. On the other hand, the decoder 306 determines whether the instruction B is a branch instruction from the input first byte, when the instruction is assumed to be a branch instruction, in this case, the control signal line 308 outputs high-level. The counter 309 holds the instruction word length (=2) of the instruction B, decrements holding value and outputs decremented value (=1) to the latch circuit 310 when the control signal line 223 becomes to be high-level.

Next, when the control signal 223 becomes to be high-level at a fetch cycle of a second byte of the instruction B, the counter 309 decrements holding value (=1) and outputs the decremented value (=0) to the latch circuit 310. As the value of the latch circuit 310 becomes to be "0", the control signal line 311 becomes to be high-level. On the other hand, as the control signal line 308 of the decoder 306 is high-level, the output signal line 316 of the two-inputs AND circuit 315 becomes to be high-level too. When the control signal line 316 becomes to be high-level, the latch circuit 317 makes the control signal line 221 in low-level. The control signal line 221 keeps outputting low-level until the control signal line 318 becomes to be low-level. As the control signal line 221 is in low-level, the fetch request signal line 217 is in high-level, as the control signal line 221 is in low-level even if the fetch request is generated, an output signal line 219 of the two-inputs AND circuit 216 keeps in low-level, a fetch request to the bus control 213 is not performed. The fetch request is not accepted until the branch destination address is decided, and the control signal line becomes to be low-level and the control signal line 221 becomes to be high-level.

As shown in an embodiment above mentioned, minimizing of unproductive

fetching is allowed by placing a pre-decoder before fetching an instruction in the processor from an external bus, detecting an instruction currently fetched is a branch instruction and the branch instruction and all byte composing the instruction are fetched in the processor so that an abortion request of fetching is output in earlier stage.

[Effect of the Invention]

As mentioned above, in the present invention, a pre-decoder is placed in a stage prior to fetch an instruction from an external data bus into an internal instruction buffer, a first byte of the instruction to be fetched is decoded, when the decoded instruction is a branch instruction, if it is detected that all bytes composing the branch instruction are fetched, an abortion-request of fetching is issued immediately to abort instruction fetching to minimize instruction fetching which becomes unproductive by execution of the branch instruction, then data access waiting time is decreased, improvement of computer processing speed can be obtained.

[4.Brief description of Drawings]

Fig.1 is a block diagram of one embodiment of the present invention.

Fig.2 is a detailed circuit diagram of a pre-decoder shown in Fig.1.

Fig.3 is a timing chart at operation of the pre-decoder shown in Fig.2.

Fig.4 is a block diagram of a CPU having traditional pre-fetch mechanism.

- 1 : external data bus
- 2 : CPU
- 3 : external data bus
- 4 : CPU
- 101, 201 : instruction bus
- 102, 202 : internal instruction buffer
- 103, 203 : queue read counter
- 104, 204 : queue write counter
- 105, 205 : instruction bus
- 106, 206 : decoder
- 107, 207 : micro-instruction bus
- 108, 208 : micro program
- 109, 209 : a group of control signals

110, 210 : execution unit
111, 221 : fetch abortion signal
113, 213 : bus control part
114, 115, 214, 215 : address data bus
116, 216 : queue control part
117, 217 : fetch request signal line
118, 218 : two-inputs AND circuit
119, 219 : fetch request signal line
220 : pre-decoder
222, 223 : fetch cycle signal line
301, 301 : gate
303, 304 : data bus
305, 306 : decoder
307 : instruction length signal line
308 : branch instruction signal line
309 : counter
310 : latch
311, 312 : control signal line
313 : two-inputs AND circuit
314 : control signal line
315 : two-inputs AND circuit
316 : control signal line
317 : latch circuit
318 : fetch re-start signal line

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- BLACK BORDERS**
- IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- FADED TEXT OR DRAWING**
- BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- SKEWED/SLANTED IMAGES**
- COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- GRAY SCALE DOCUMENTS**
- LINES OR MARKS ON ORIGINAL DOCUMENT**
- REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- OTHER:** _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.